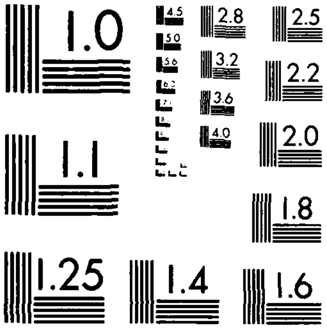
AD-A195 685 REPRESENTATION FOR NARRATIVE TEXT(U) JAYCOR ALEXANDRIA 1/1
VA 12 MAY 88 N00014-85-C-2444

UNCLASSIFIED F/G 12/5 NL


1.0
4.5
5.0
5.6
6.3
7.1
8
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

REPRESENTATION FOR NARRATIVE TEXT

Deliverable No.: B002

Prepared by:
JAYCOR

Prepared for:
Naval Research Laboratory
Washington, DC 20375-5000

In Response to:
Contract #N00014-85-C-2444

12 May 1988

Introduction

This technical report fulfills the report deliverable described in paragraph B002 of form 1423 of contract N00014-86-C-2444. The ~~following~~ report lists and describes the new linguistic information which has been implemented in the TERSE system. The software for this deliverable has been delivered and is present on the Symbolics workstation at NRL. All files and knowledge bases for the system can be found in the directory "SY2>:terse>".

1. The Basic Representation - Information Formats

This section describes the basic implementation of information formats in the TERSE system. The next section will distinguish which items were added to the representation to fulfill the deliverable requirements.

The TERSE system contains a knowledge base called **datastructures** that contains all the definitions for the structures that hold information formats. In addition to this, there is one file of lisp code which is called "makefacts.lisp". This file contains the code to begin processing a file containing a listified representation of information formats. One message per file read in from this file then other procedures attached to the actual data structures are called from this top level routine. In the future, if the representation produced by the parser is changed, these routines must be accordingly modified. All the procedures that are attached to the structures in the knowledge base **datastructures** are implemented as methods and are always named "fill-in-slots" although the code for each particular structure element is different. This approach takes advantage of the *object oriented* methods of programming available in KEE.

1.1. A Description of Datastructure Items

Information formats are basically made up of *formats, connectives, semantic category entries.* All of these elements explicitly appear in the following diagram of the KEE knowledge base **datastructures**. However, in the knowledge base there are some additional elements. These elements are arranged in conjunction with the standard formats, connectives, and entries in a hierarchical manner. They are introduced into the representation to house information which is common to all structures that fall beneath it in the inheritance hierarchy. This approach is consistent with frame based programming. The following section describes what information is housed at each frame and a brief explanation to describe the purpose of the information if necessary. The level of indentation indicates the inheritance of information in the hierarchy.

> **temp-units** - all elements generated from the reading of a message or during inferencing inherit the slots of this frame. It contains a slot named *inferred?* which contains a boolean flag indicating whether a unit was inferred or was provided by the parser. This is needed to prevent units which are inferred from being picked as the "summary". Two other slots containing methods are present at this frame. The method *delete.inferred.members* performs the function of deleting any member units that have the *inferred?* flag set to true. This is useful when debugging the system because it enables the programmer to rerun the system without having to read in the message all over again. The method *delete.members* simply retrieves all the units that are descended from the unit to which this message is sent and deletes them.

entry - this frame contains all of the modifier slots needed to hold modifiers for a semantic category entry. The following modifying slots are defined for the frame: *code, det, evid, modal, neg, quant, quant-mod.* The word which was actually classified as a particular semantic category is contained in the slot named *head.* The slot *text* contains the entire text of the unit including all the modifying words as a text string. The only other slot defined at this level is *headcat.* This slot is used to hold special categorizing tags generated during inferencing.

admin - unit to contain entries of semantic category *admin.* No additional slots are present at this unit. All slots are inherited.

area - unit to contain entries of semantic category *area.* No additional slots are present at this unit. All slots are inherited.

assist - unit to contain entries of semantic category *assist.* No additional slots are present at this unit. All slots are inherited.

func - unit to contain entries of semantic category *func.* No additional slots are present at this unit. All slots are inherited.

invest - unit to contain entries of semantic category *invest.* No additional slots are present at this unit. All slots are inherited.

msg - unit to contain entries of semantic category *msg.* No additional slots are present at this unit. All slots are inherited.

org - unit to contain entries of semantic category *org.* No additional slots are present at this unit. All slots are inherited.

partloc - unit to contain entries of semantic category *partloc.* No additional slots are present at this unit. All slots are inherited.

piece - unit to contain entries of semantic category *piece.* No additional slots are present at this unit. All slots are inherited.

process - unit to contain entries of semantic category *process.* No additional slots are present at this unit. All slots are inherited.

procure - unit to contain entries of semantic category *procure.* No additional slots are present at this unit. All slots are inherited.

repair - unit to contain entries of semantic category *repair.* No additional slots are present at this unit. All slots are inherited.

stask - unit to contain entries of semantic category *stask.* No additional slots are present at this unit. All slots are inherited.

status - unit to contain entries of semantic category *status.* No additional slots are present at this unit. All slots are inherited.

substance - unit to contain entries of semantic category *substance.* No additional slots are present at this unit. All slots are inherited.

op - this unit contains information which is of identical structure to the frame used to hold semantic categories but does not contain a semantic category element. It is used to house the head information for connective structures. An additional slot is introduced for this unit. The slot's name is *op-type.* Connectives are now subclassified and this slot contains the subclassification.

time-entries - this unit is used to distinguish the elements that fall beneath it as structures used to hold temporal information and not semantic category information. The structure of the information provided in these time modifiers is identical to the structure of the semantic category information so it is convenient to make use of the existing **entry** definition for this frame.

change - a unit containing temporal information.

beg - a unit containing temporal information.

end - a unit containing temporal information.

model-objects - this unit adds two additional slots to the semantic category elements that are descended from it. One slot is called *referent* and is used to contain a pointer to the actual piece of equipment that the dereferencing procedures determines to be the referent of the nominal. The second slot is named *state* and is used during inferencing to indicate what state the model object has been found or inferred to be in. The third slot is the *match* slot. It is used to contain the proposed matches to objects in the domain model during nominal dereferencing.

falarm - unit to contain entries of semantic category *alarm.* No additional slots are present at this unit. All slots are inherited.

part - unit to contain entries of semantic category *part.* No additional slots are present at this unit. All slots are inherited.

prop - unit to contain entries of semantic category *prop.* No additional slots are present at this unit. All slots are inherited.

match - this unit contains two additional slots. Each instance of this unit (created during dereferencing) represents a possible referent of a nominal. One slot called *object* contains the actual pointer to an object in the domain model. The other slot called *score* contains a score which is computed during disambiguation. The match with the largest score is picked as the referent for the nominal when nominal processing is completed.

metaformat - this unit contains slots which are common to both format and connective structures. The *explanation* slot is used to collect english, canned explanations contributed by rules when they fire on a particular format or connective. The *score* slot accumulates the points assigned by rules as they fire on these structures. The *sentence.no* slot is used to keep track of which sentence the structure was derived from and also to keep a history of the order of the original discourse. The *text* slot contains the text string which represents each connective or format. This text is displayed in the user interface.

conn - this unit represents the basic connective is information formats. It contains three additional slots. The first slot, *op* contains another structure of type **op** which contains the word which actually performs the function of connecting. The slots *arg1* and *arg2* each contain a structure of type **format** or **conn.** These represent the two arguments that are being connected by the *op.*

format - this unit contains a slot for every possible semantic category: *admin, area, assist, func, invest, alarm, part, prop, msg, org, partloc, piece, process, procure, repair, stask, status*, and *substance.* Also at this level, several other slots are introduced. *create-format-images!* is a method which adds a display box to the user interface in which the text of the format is displayed. *explanation-window-name* contains the name of the window that displays the explanations attributed to that format. This value is accessed by the function which opens and closes the window on user demand. *fill-in-slots* is a method which takes a listified format and translates it into the appropriate KEE units. *format-window-name* contains the name of the window which displays the format text. *format.no* is a slot that contains a numeric value representing the format count up to and including the current format. This number is used in generating names for other units, primarily. The slots *t-beg, t-change, t-end, t-prep1, t-prep2, t-refpt1, t-refpt2, t-unit1*, and *t-unit2* are all used to contain temporal information. *t-vtense* contains the verb tense used in the original sentence if any.

2. Summary

For this deliverable, we have exhaustively encoded all linguistic information passed to us from the parser. The representation has been implemented in KEE and thus is compatible with all other knowledge bases that have been implemented for this system. The representation is still *information formats* although they are now implemented in KEE. Information formats are extensible to new message domains in that the basic concept behind the representation remains the same, only the actual semantic categories may change. In order to modify this representation, it is necessary to replace the appropriate semantic category units with the new ones for the new domain and also to modify the slots of the *format* unit accordingly. It will also be necessary to alter a simple list of the acceptable semantic category names in the lisp method *fill-in-slots* attached to the *format* unit.

Several new pieces of information have been encoded in this implementation of the representation. In particular, information formats have been embellished with some temporal information. In particular, the slots *vtense, t-change, t-beg, t-end, t-prep1, t-unit1, t-refpt1, t-prep2, t-unit2, t-refpt2* have been added.

Several new semantic categories were added to this implementation. They include *alarm, area, partloc, piece,* and *substance. Connectives* are now subclassified into one of the following subclasses: *conjoined, time-conj, relation, sub-conj, embedded, rel-clause.* The semantic category entries have a new modifier called *evid.* This constitutes all the deeper linguistic information that has been provided from the parser.

FORMAT

ADMIN	ALARM	AREA	ASSIST	FUNC	INVEST	MSG	ORG	PART	PARTLOC	PIECE	PROCESS	PROCURE	PROP	REPAIR	REQUIRE	STASK	STATUS	SUBSTANCE

FORMAT MODIFIER

TIME				EVNT-TIME					
VTENSE	CHANGE	BEG	END	TPREP1			TPREP2		
				PREP	TM-UNIT	REFPT	PREP	TM-UNIT	REFPT

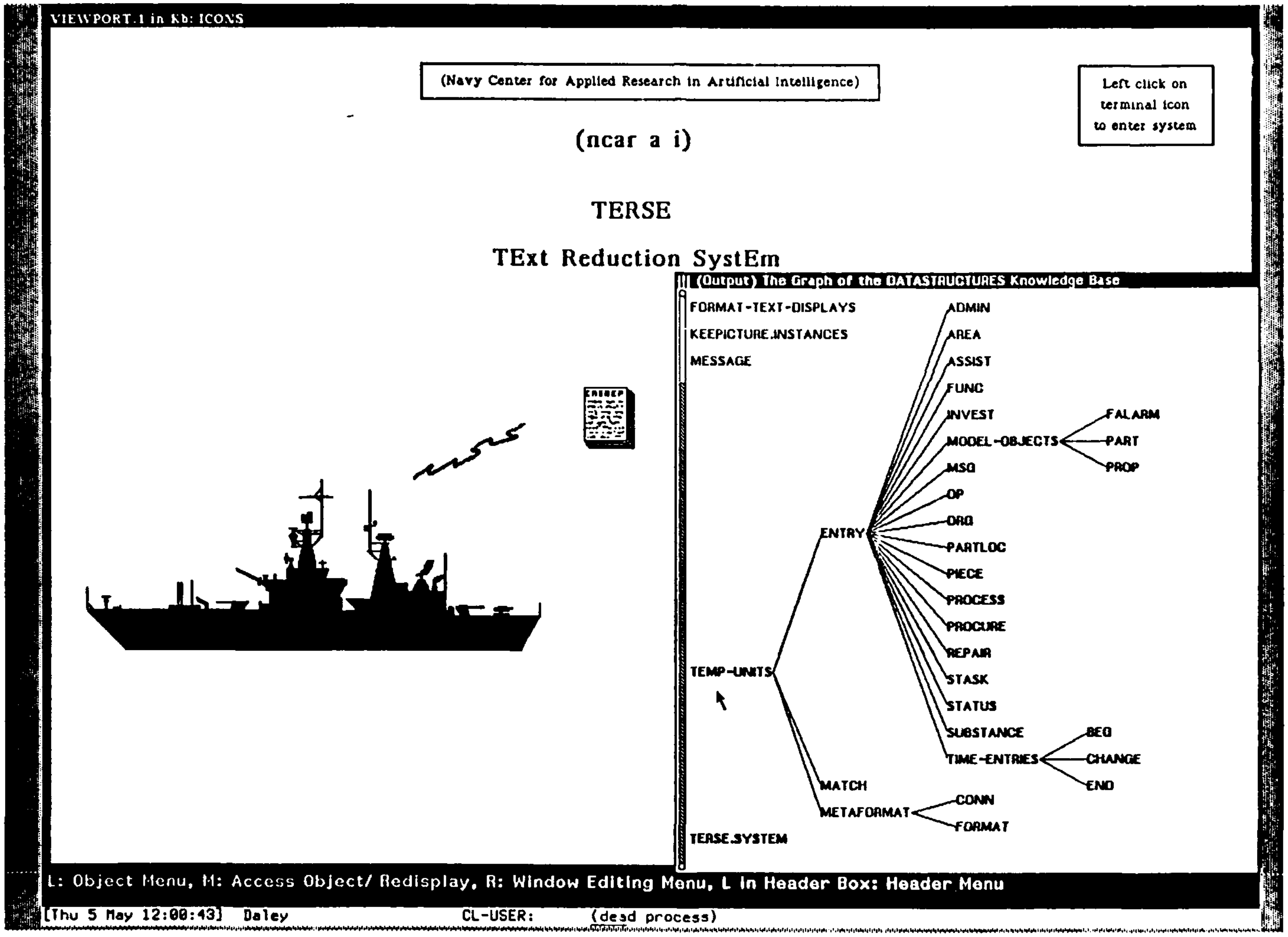
SEMANTIC CATEGORY MODIFIERS

NEG
QUANT
DET
EVID
MODAL
HEAD-ENTRY
TEXT

CONNECTIVE

CONJOINED	TIME-CONJ	RELATION	SUB-CONJ	EMBEDDED	REL-CLAUSE

NEW FORMAT



VIEWPORT.1 in Kb: ICONS

(Navy Center for Applied Research in Artificial Intelligence)

Left click on terminal icon to enter system

(ncar a i)

TERSE

TExt Reduction SystEm

(Output) The Graph of the DATASTRUCTURES Knowledge Base

L: Object Menu, M: Access Object/ Redisplay, R: Window Editing Menu, L in Header Box: Header Menu

[Thu 5 May 12:00:43] Daley CL-USER: (dead process)

END
DATED
FILM
9–88
DTIC